

## Operating System and Design (19CS2106A) Advanced Lab- 5

### XV6 file system checker:

For this project, we will use the xv6 file system image as the basic image that we will be reading and checking. The file **include/fs.h** includes the basic structures you need to understand, including the superblock, on disk inode format (struct dinode), and directory entry format (struct dirent). The tool **tools/mkfs.c** will also be useful to look at, in order to see how an empty file-system image is created.

Make sure to look at **fs.img**, which is a file system image created when you make xv6 by the tool **mkfs** (found in the tools/ directory of xv6). The output of this tool is the file **fs.img** and it is a consistent file-system image


Steps:

1. \$ git clone <https://github.com/gauthamsunjay/filesystemchecker.git>
2. \$ cd filesystemchecker
3. \$ ls  
fs.h README.md xcheck.c
4. \$ cc xcheck.c
5. \$ git clone git://github.com/mit-pdos/xv6-public.git xv6
6. \$ cd xv6
7. \$ make qemu-nox
8. type ctrl a, x
9. \$ cd ..
10. \$ cp fs.h ./xv6
11. \$ cp xcheck.c ./xv6
12. \$ cd xv6
13. \$ cc xcheck.c
14. \$ ls

When we type ls we can see fs.img file which has been created

15. \$ ./a.out fs.img

It prints "ERROR:address used by inode but marked free in bitmap" as one inode is not enabled.

 osd-190031187@team-osd:~/filesystemchecker/xv6

```
[osd-190031187@team-osd xv6]$ ./a.out fs.img
ERROR: address used by inode but marked free in bitmap.
[osd-190031187@team-osd xv6]$
```

## UNIX system programming

### 1. dup2

The dup() system call creates a copy of a file descriptor.

- It uses the lowest-numbered unused descriptor for the new descriptor.
- If the copy is successfully created, then the original and copy file descriptors may be used interchangeably.
- They both refer to the same open file description and thus share file offset and file status flags.

### dup.c

osd-190031187@team-osd:~/xv6-public

```
GNU nano 2.3.1 File: dup.c
#include <stdio.h>
#include<unistd.h>
#include<fcntl.h>

int main()
{
    int fd=open("dup.txt",O_WRONLY|O_APPEND);
    if(fd<0)
        printf("error opening the file\n");
    int cpfd=dup(fd);
    write(cpfd,"this will be the output to the file named dup.txt\n",46);
    write(fd,"this will also be the output to the file named dup.txt\n",51);
    return 0;
}
```

### dup.txt:before running prog

osd-190031187@team-osd:~/xv6-public

```
GNU nano 2.3.1 File: dup.txt
i my id is 190031187
```

osd-190031187@team-osd:~/xv6-public

```
[osd-190031187@team-osd xv6-public]$ nano dup.c
[osd-190031187@team-osd xv6-public]$ nano dup.txt
[osd-190031187@team-osd xv6-public]$ nano dup.txt
[osd-190031187@team-osd xv6-public]$ gcc dup.c
[osd-190031187@team-osd xv6-public]$ ./a.out
[osd-190031187@team-osd xv6-public]$ nano dup.txt
[osd-190031187@team-osd xv6-public]$
```

After running program : dup .txt consists

The screenshot shows a terminal window with the title bar 'GNU nano 2.3.1' and 'File: dup.txt'. The content of the file is: 

```
i my id is 190031187
this will be the output to the file named dup.this will also be the output to the file named dup.
```

int dup2(int oldfd, int newfd); oldfd: old file descriptor newfd new file descriptor which is used by dup2() to create a copy.

dup2.c

The screenshot shows a terminal window with the title bar 'GNU nano 2.3.1' and 'File: dup2.c'. The content of the file is: 

```
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <fcntl.h>

int main()
{
    int fd=open("tricky.txt",O_WRONLY,O_APPEND);
    dup2(fd,1);
    printf("I WILL BE PRINTED IN THE FILE TRICKY.TXT\n");
    return 0;
}
```

tricky.txt:before running prog

The screenshot shows a terminal window with the title bar 'GNU nano 2.3.1' and 'File: tricky.txt'. The content of the file is: 

```
hi i am tricky.txt my id is 190031187
```

After running program: tricky .txt consists

The screenshot shows a terminal window with the title bar 'GNU nano 2.3.1' and 'File: tricky.txt'. The content of the file is: 

```
I WILL BE PRINTED IN THE FILE TRICKY.TXT
```

## 2. Unix sockets: TCP, UDP,

It is set to the protocol used. Internet services are provided using either TCP or UDP.

TCP:

DISPLAYS ALL TCP SOCKETS USING: `$ss -t -a` command

```
osd-190031187@team-osd:~/filesystemchecker/xv6
[osd-190031187@team-osd xv6]$ ss -t -a
State      Recv-Q Send-Q           Local Address:Port           Peer Address:Port
LISTEN     0      50              *:mysql                       *:*
LISTEN     0      5              *:5901                         *:*
LISTEN     0      128             *:sunrpc                      *:*
LISTEN     0      5              *:5905                         *:*
LISTEN     0      128             *:6001                         *:*
LISTEN     0      128             *:6005                         *:*
LISTEN     0      5              192.168.122.1:domain         *:*
LISTEN     0      128             127.0.0.1:ipp                *:*
LISTEN     0      32              *:920                          *:*
LISTEN     0      100             127.0.0.1:smtp               *:*
LISTEN     0      128             *:985                          *:*
ESTAB      0      0              192.168.2.52:985             160.238.73.26:20754
ESTAB      0      0              192.168.2.52:985             103.96.18.28:51127
ESTAB      0      0              192.168.2.52:985             49.37.150.189:56387
ESTAB      0      0              192.168.2.52:985             175.101.68.113:61331
ESTAB      0      0              192.168.2.52:985             45.249.75.162:49486
ESTAB      0      0              192.168.2.52:985             43.228.95.113:54085
ESTAB      0      560             192.168.2.52:985             49.37.150.140:50929
ESTAB      0      0              192.168.2.52:985             223.196.168.203:9939
ESTAB      0      0              192.168.2.52:985             146.196.36.7:50432
ESTAB      0      0              192.168.2.52:985             183.82.158.228:50196
ESTAB      0      0              192.168.2.52:985             106.0.39.210:26466
ESTAB      0      0              192.168.2.52:985             106.217.223.232:63226
ESTAB      0      0              192.168.2.52:985             183.82.157.62:21857
ESTAB      0      0              192.168.2.52:985             59.96.179.192:50889
ESTAB      0      0              192.168.2.52:985             160.238.75.212:7267
ESTAB      0      0              192.168.2.52:985             43.225.21.90:50780
ESTAB      0      0              192.168.2.52:985             103.49.206.209:58077
ESTAB      0      0              192.168.2.52:985             202.3.72.107:50565
ESTAB      0      0              192.168.2.52:985             175.101.106.50:50047
ESTAB      0      64              192.168.2.52:985             157.48.207.149:50407
ESTAB      0      208             192.168.2.52:985             117.192.181.105:55790
ESTAB      0      0              192.168.2.52:985             192.140.155.201:56841
ESTAB      0      0              192.168.2.52:985             117.194.91.74:9439
ESTAB      0      0              192.168.2.52:985             160.238.75.248:52985
ESTAB      0      0              192.168.2.52:985             175.101.106.11:55262
ESTAB      0      0              192.168.2.52:985             223.230.34.217:53535
ESTAB      0      0              192.168.2.52:985             157.42.23.100:50418
ESTAB      0      0              192.168.2.52:985             223.237.6.175:62350
ESTAB      0      0              192.168.2.52:985             157.44.112.10:51409
ESTAB      0      0              192.168.2.52:985             157.48.186.101:61759
ESTAB      0      0              192.168.2.52:985             49.37.132.216:52987
ESTAB      0      0              192.168.2.52:985             27.49.80.22:64092
ESTAB      0      0              192.168.2.52:985             103.96.17.72:64706
```

DISPLAYS ALL UDP SOCKETS USING: `$ss -u -a` command

```
osd-190031187@team-osd:~/filesystemchecker/xv6
[osd-190031187@team-osd xv6]$ ss -u -a
State      Recv-Q Send-Q           Local Address:Port           Peer Address:Port
UNCONN     0      0              *:13619                       *:*
UNCONN     0      0              192.168.122.1:domain         *:*
UNCONN     0      0              *:virbr0:bootps              *:*
UNCONN     0      0              *:sunrpc                      *:*
UNCONN     0      0              127.0.0.1:323                *:*
UNCONN     0      0              *:971                         *:*
UNCONN     0      0              *:mdns                       *:*
UNCONN     0      0              [::]:sunrpc                  [::]:*
UNCONN     0      0              [::1]:323                    [::]:*
UNCONN     0      0              [::]:971                     [::]:*
```

## 3. Demonstrate How RPC Works: invoking square remote procedure with arguments. Invoking date remote procedure without argument

A remote procedure call is an interprocess communication technique that is used for client-server based applications. It is also known as a subroutine call or a function call.

A client has a request message that the RPC translates and sends to the server. This request may be a procedure or a function call to a remote server. When the server receives the request, it sends the required response back to the client. The client is blocked while the server is processing the call and only resumed execution after the server is finished.

Some of the advantages of RPC are as follows –

- Remote procedure calls support process oriented and thread oriented models.
- The internal message passing mechanism of RPC is hidden from the user.
- The effort to re-write and re-develop the code is minimum in remote procedure calls.
- Remote procedure calls can be used in distributed environment as well as the local environment.

• Many of the protocol layers are omitted by RPC to improve performance.

**Disadvantages of Remote Procedure Call**

Some of the disadvantages of RPC are as follows –

- The remote procedure call is a concept that can be implemented in different ways. It is not a standard.
- There is no flexibility in RPC for hardware architecture. It is only interaction based.
- There is an increase in costs because of remote procedure call.